УДК 004.04

**ПРОГРАММНАЯ ПОДСИСТЕМА ТЕСТИРОВАНИЯ ЗНАНИЙ ЯЗЫКОВ ОПИСАНИЯ АППАРАТУРЫ**

**С.В. Астахов** fzastahov@gmail.com

**Н.В. Лапшин** nikita.lapshin2000@gmail.com

**Т.А. Ким** 1234@mail.ru

**МГТУ им. Н.Э. Баумана, Москва, Российская Федерация**

|  |  |
| --- | --- |
| **Аннотация** | **Ключевые слова** |
| *Статья посвящена разработке программной подсистемы тестирования знаний языков описания аппаратуры, которая предоставляет возможности по управлению учебными материалами и автоматической проверке заданий, в том числе, заданий на описание аппаратных устройств на языке Verilog. Проведен анализ существующих систем тестирования знаний, в ходе анализа сформулированы функциональные требования и составлена диаграмма вариантов использования программной подсистемы тестирования знаний языков описания аппаратуры. Спроектирована архитектура и компоненты подсистемы. Проведено функциональное и нагрузочное тестирование разработанной подсистемы.* | *Тестирование знаний, язык описания аппаратуры, HDL, Verilog, система дистанционного обучения, образовательный портал.* |

**Введение.** В настоящее время существует огромное количество образовательных ресурсов, посвященных тематике информационных технологий. Несмотря на это, на данный момент в открытом доступе наблюдается дефицит ресурсов, посвященных изучению языков описания аппаратуры. Среди существующих образовательных платформ есть лишь несколько таких, на которых возможно настроить автоматическую проверку заданий на написание исходного кода на языке Verilog (или каком-либо другом языке описания аппаратуры) непосредственно в рамках веб-приложения. При этом процесс настройки весьма сложен, поэтому авторы курсов редко используют описанную возможность.

В настоящей статье рассмотрен процесс разработки программной подсистемы тестирования знаний языков описания аппаратуры, которая предоставляет возможности по управлению учебными материалами и автоматической проверке заданий (в том числе, заданий на описание аппаратных устройств на языке Verilog).

**Анализ существующих систем тестирования знаний.** В ходе анализа существующих систем тестирования знаний, таких как Huawei University, Coursera, Stepik и Moodle авторами была предложена классификация методов тестирования знаний (таблица 1), составленная на основе классификации методов тестирования знаний, применяемых в системе Moodle [1].

*Таблица 1*

**Классификация методов тестирования знаний**

|  |  |  |
| --- | --- | --- |
| **№** | **Тип** | **Подтип** |
| 1 | Тестирование с ответом в закрытой форме | 1.1 Выбор одного ответа  1.2 Выбор множественных ответов  1.3 Сопоставление |
| 2 | Тестирование с коротким ответом | 2.1 С автоматизированной проверкой  2.2 С проверкой преподавателем  2.3 С перекрестной проверкой |
| 3 | Тестирование с ответом в форме эссе | 3.1 С проверкой преподавателем  3.2 С перекрестной проверкой |
| 4 | Тестирование на написание исходного кода | 4.1 С проверкой по референсным значениям  4.2 Автоматизированное тестирование на проверяющей стороне  4.3 Другие |

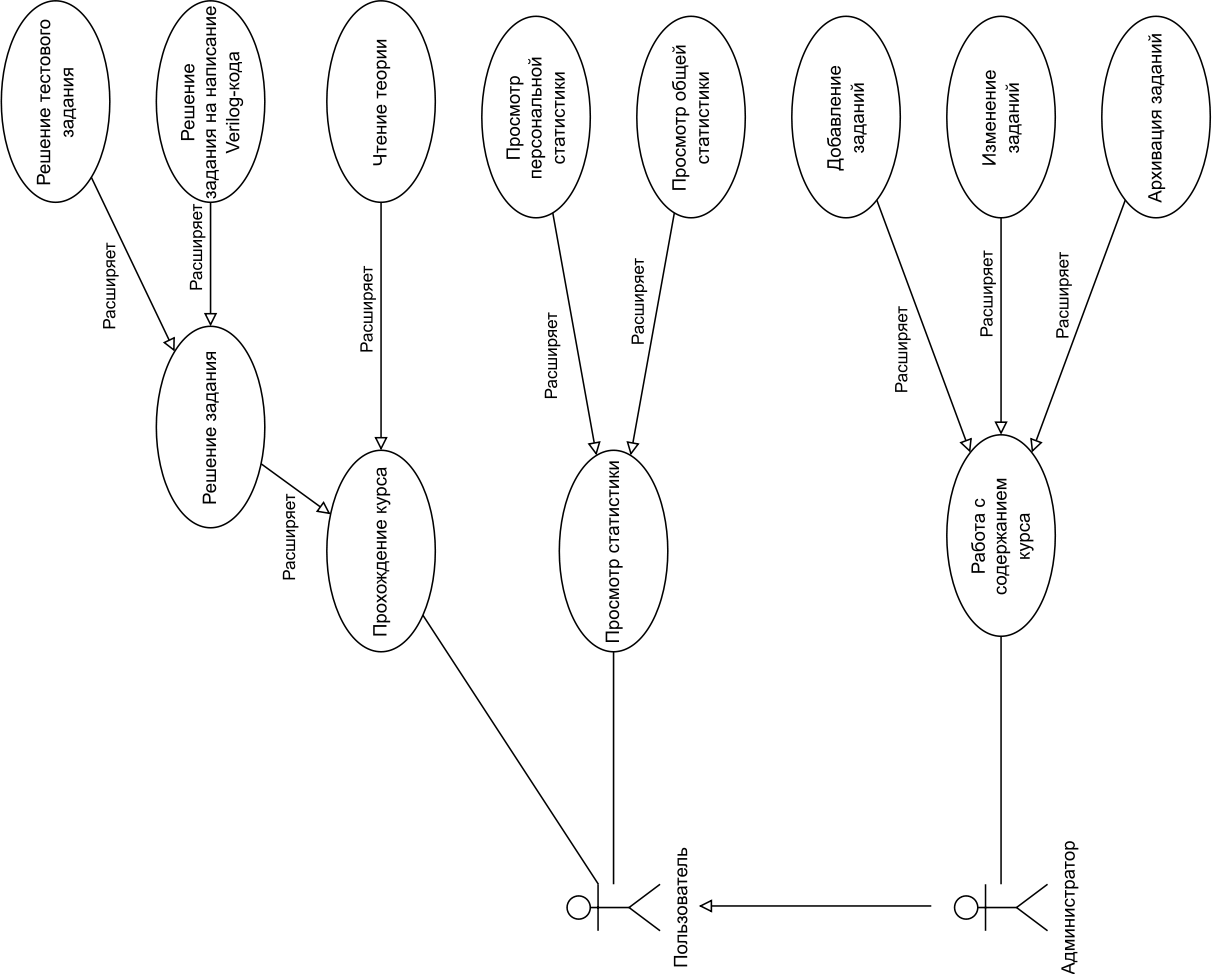
В последующем, из рассмотренных методов тестирования знаний знаний были выделены наиболее подходящие для использования в подсистеме тестирования знаний языков описания аппаратуры методы. Кроме того, были предложены типы обратной связи, предоставляемой обучающемуся, в случае допущения им ошибки при решении задания (таблица 2).

*Таблица 2*

**Методы тестирования знаний в разработанной подсистеме**

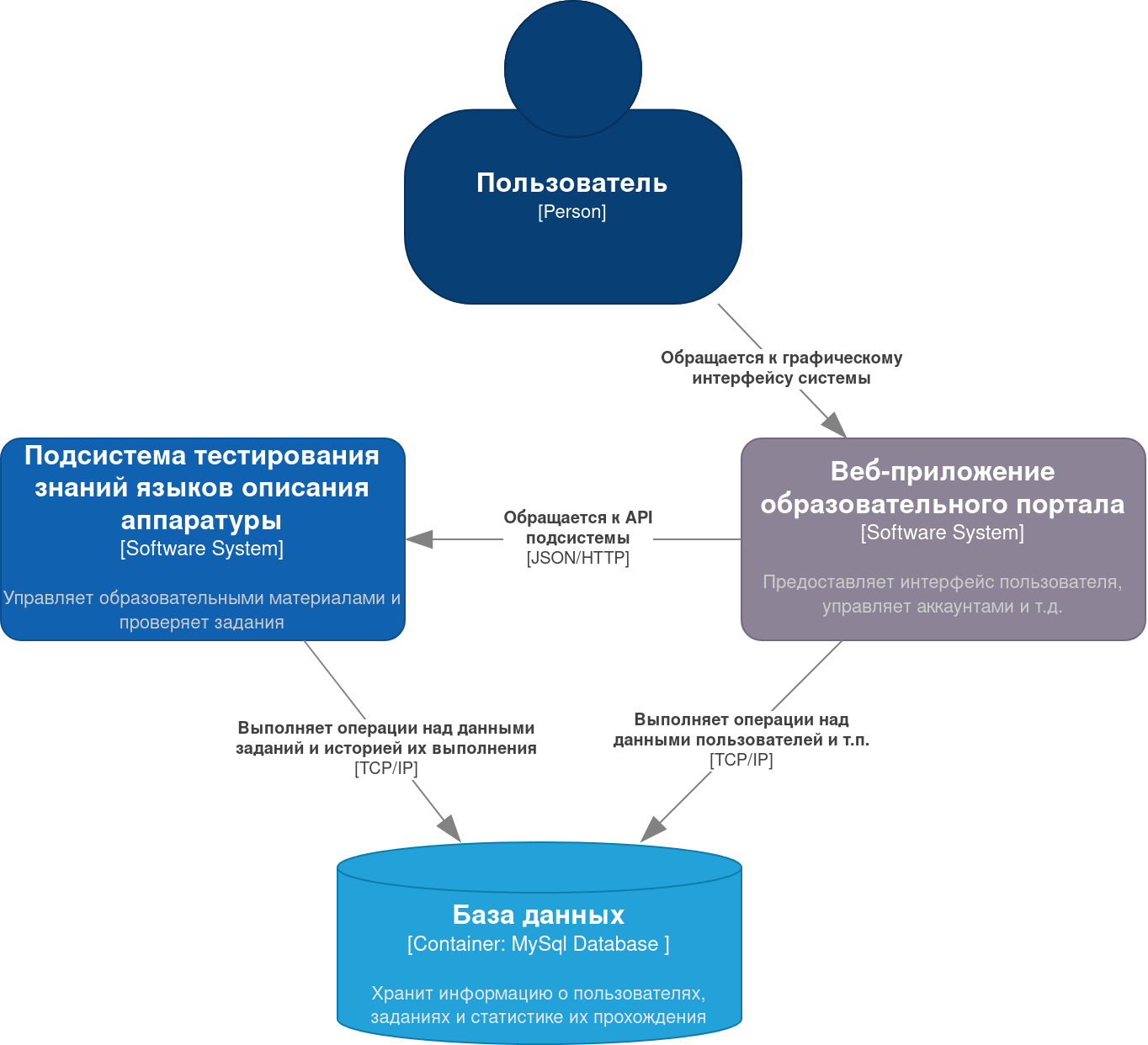
|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Тип** | **Подтип** | **Вид обратной связи** |
| 1 | Тестирование с ответом в закрытой форме | Выбор одного ответа | Текстовое пояснение ошибки |
| Выбор нескольких ответов | Информации о наличии ложноположительных  (ложноотрицательных) ответов |
| 2 | Задание на написание исходного кода | Автоматизированное тестирование на проверяющей стороне | Информация о несоответствующих сигналах |

Кроме того, в результате проведенного анализа была составлена диаграмма вариантов использования подсистемы тестирования знаний языков описания аппаратуры, представленная на рисунке 1 [2].



**Рис. 1.** Диаграмма вариантов использования подсистемы

**Проектирование архитектуры подсистемы.** После анализа аналогов необходимым этапом является разработка архитектуры. Разработанную подсистему предполагается использовать как информационной системы образовательного портала, что отражено в архитектуре информационной системы, показанной на контекст-диаграмме (нотация С4) на рисунке 2.



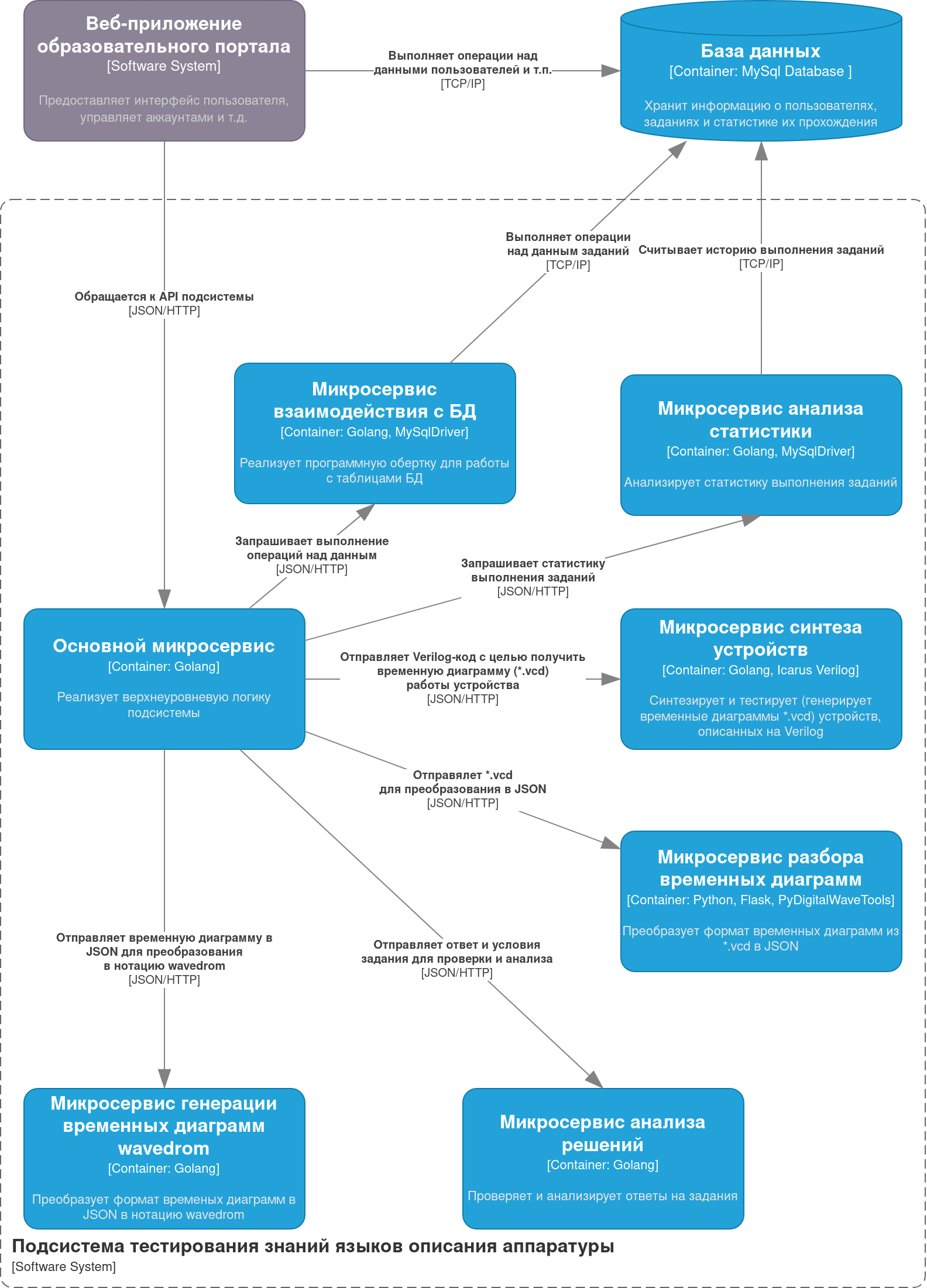
**Рис. 2.** Контекст-диаграмме информационной системы

Так как разработанная подсистема является весьма сложной, выполняемые в ней операции разнородны, могут потребовать использования различных языков и библиотек, а также некоторые из них могут занимать значительное время, при разработке было решено использовать микросервисную архитектуру.

На основе функциональных требований и представленной ранее диаграммы вариантов использования была разработана структура компонентов подсистемы, включающая следующие микросервисы:

* микросервис взаимодействия с БД — реализует CRUD-операции над данными в БД;
* микросервис анализа решений (анализатор) — выполняет проверку и анализ пользовательских решений;
* микросервис синтеза устройств (синтезатор) — выполняет синтез устройств из Verilog-кода и симулирует их работу;
* микросервис разбора временных диаграмм, микросервис генерации временных диаграмм wavedrom — преобразуют временные диаграммы в удобные для хранения и обработки форматы;
* микросервис анализа статистики;
* основной микросервис — реализует верхнеуровневую логику подсистемы, связывает остальные микросервисы.

Детализированная архитектура разработанной подсистемы показана на контейнер-диаграмме (нотация С4) на рисунке 2.



**Рис. 3.** Контейнер-диаграмма разработанной подсистемы

**Проектирование базы данных.** После проектирование архитектуры подсистемы была спроектирована структура ее базы данных. В результате анализа предметной области удалось выделить описанные ниже сущности.

Сущность «Задание» — содержит информацию о порядковом номере задания, его условиях, правильном ответе, цене в баллах и т.п.;

Сущность «Пользователь» — позволяет идентифицировать пользователя по ID, узнать, обладает ли пользователь правами администратора и узнать его псевдоним (т.н. «никнейм»). Кроме того, эта сущность может нести в себе дополнительную информацию, необходимую веб-приложению образовательного портала.

Сущность «Попытка решения» — содержит информацию, об успешности и времени каждой попытки решения задания каким-либо пользователем.

Полученная даталогическая схема базы данных в нотации Мартина изображена на рисунке 4.



**Рис. 4.** Даталогическая схема базы данных

**Разработка микросервиса синтеза устройств.**